

Automatic Keyword Extraction with Text Clustering by KNN Algorithm and AHC Algorithm considering Feature Similarities

Taeho Jo
President
Alpha AI Research
Cheongju, South Korea
tjo018@naver.com

Abstract—In this research, we propose the full automation of the keyword extraction by introducing the text clustering. In the previous work, even if the KNN algorithm which considers the feature similarities was already applied for the keyword extraction, we must determine a category or a domain, in advance. In this research, text clusters are constructed based on their contents, and keywords from an input text are extracted by a classifier which corresponds to its most similar cluster. The AHC algorithm and the KNN algorithm which consider the feature similarities are adopted respectively to the approaches to the text clustering and the keyword extraction. The goals of this research are to automate the keyword extraction and to improve the performance of both tasks, using the modernized KNN version and the AHC version.

I. INTRODUCTION

The keyword extraction is defined as the process of extracting important words from a text with respect to its contents. The task which is covered in this research is interpreted into a binary classification of words into keyword or non-keyword. The process of keyword extraction is to index a text into words, classify each word into one of the two categories, and extract ones which are classified into keyword. The binary classification which is mapped from the keyword extraction is given as a domain dependent classification where a same word may be classified differently depending on the domain. When the input text is given as an extraction source, we need to present its domain.

This research is motivated by the requirement of pre-defining domains for implementing the keyword extraction system. The keyword extraction is defined as an independent binary classification domain by domain. Domains should be decided manually as a list or a tree for designing the keyword extraction system. The results from applying the KNN algorithm and the AHC algorithm which consider the feature similarities to the keyword extraction and the text clustering respectively were successful. The process of deciding domains is automated by connecting the keyword extraction with the text clustering.

The idea of this research is to apply the KNN algorithm and the AHC algorithm which consider the feature similarities to the keyword extraction and the text clustering, respectively. The similarity metric between two numerical vectors

which considers the feature similarities is defined, and the KNN algorithm and the AHC algorithm are modified based on the similarity metric. The KNN version and the AHC version are used respectively for the keyword extraction and the text clustering. The domains are defined through text clustering automatically and the keyword extraction is executed domain by domain. The domain is decided to the input text based on its similarities with cluster prototypes, and a classifier which corresponds to the domain is nominated.

Let us mention some benefits from this research. As the solution to the poor discrimination among sparse vectors, the feature similarities are considered in computing the similarity between numerical vectors. The performances in the keyword extraction and the text clustering are improved by adopting the modified versions of the KNN algorithm and the AHC algorithm. The domains are automatically predefined by connecting the keyword extraction with the text clustering. A domain is automatically decided to an input text from which keywords are extracted.

Let us mention some remaining tasks as the further discussions on this research. The feature similarities among some features, rather than all features, should be computed for reducing the complexity. More advanced machine learning algorithms and deep learning algorithms need to be modified into the proposed versions. The keyword extraction system which relates to the text clustering should be implemented as a real program or a library. We need to implement the circular cooperation between the keyword extraction and the text classification for reinforcing both tasks mutually.

II. PROPOSED APPROACH

A. Similarity Metric

This section is concerned with the process of encoding a word into a numerical vector. Texts are used as features for encoding a word so, and some are selected from a text collection. A similarity between texts is computed based on their shared words, and it is used as a similarity between features. The similarities among features are considered for computing the semantic similarity between words. In this section, we describe the process of encoding a word into a

numerical vector and the process of computing the similarity between words.

The process of encoding words into numerical vectors is illustrated in Figure 1. Texts are gathered as a collection from external sources as feature candidates. Only some texts are selected by a criterion among the gathered texts. In each word, its weights in the texts are computed as elements of the numerical vector. Refer to [3] for studying the process and the selection criteria.

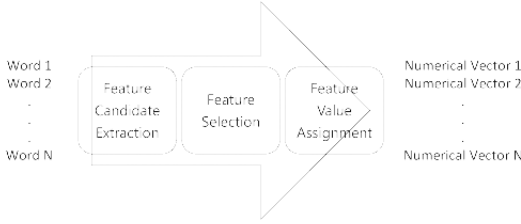


Figure 1. Process of Encoding Words into Numerical Vectors

The frame of computing the similarity between two numerical vectors is illustrated in Figure 2. The two d dimensional vectors and the d features are respectively notated respectively by $\mathbf{x} = [x_1 \ x_2 \ \dots \ x_d]$, $\mathbf{y} = [y_1 \ y_2 \ \dots \ y_d]$, and $f_1, f_2, \dots, f_d$. The feature similarity refers to the similarity between two features, which is notated by $\text{sim}(f_1, f_2)$. The feature value similarity is the similarity between two vectors, $\mathbf{x}$ and $\mathbf{y}$, such as cosine similarity. The similarity between words is computed by considering the two kinds of similarities.

$$\begin{aligned} \mathbf{x} &= [x_1, x_2, \dots, x_d] \\ \mathbf{y} &= [y_1, y_2, \dots, y_d] \end{aligned} \quad \begin{array}{l} \text{Feature} \\ \text{Similarity} \\ \\ \text{Feature} \\ \text{Value} \\ \text{Similarity} \end{array}$$

Figure 2. Frame of Computing Similarity between Numerical Vectors

The frame of computing the similarity between two numerical vectors is illustrated in Figure 2. The two d dimensional vectors and the d features are respectively notated respectively by $\mathbf{x} = [x_1 \ x_2 \ \dots \ x_d]$, $\mathbf{y} = [y_1 \ y_2 \ \dots \ y_d]$, and $f_1, f_2, \dots, f_d$. The feature similarity refers to the similarity between two features, which is notated by $\text{sim}(f_1, f_2)$. The feature value similarity is the similarity between two vectors, $\mathbf{x}$ and $\mathbf{y}$, such as cosine similarity. The similarity between words is computed by considering the two kinds of similarities.

Let us mention the process of computing the similarity between numerical vectors as the semantic similarity between words. The similarity between features, f_i and f_j , $\text{sim}(f_i, f_j)$, is abbreviated into s_{ij} . The similarity between

features is computed by equation (1),

$$s_{ij} = \frac{2 \times |D_i \cap D_j|}{|D_i| + |D_j|} \quad (1)$$

where D_i is the set of words in the feature, f_i , and D_j is the set of words in the feature, f_j , and the similarity matrix with the d features is constructed as a $d \times d$ square matrix, as shown in equation (2),

$$\mathbf{S} = \begin{pmatrix} s_{11} & s_{12} & \dots & s_{1d} \\ s_{21} & s_{22} & \dots & s_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ s_{d1} & s_{d2} & \dots & s_{dd} \end{pmatrix}. \quad (2)$$

The similarity between two numerical vectors, $\mathbf{x}$ and $\mathbf{y}$, is computed by equation (3),

$$\text{sim}(\mathbf{x}, \mathbf{y}) = \frac{\sum_{i=1}^d \sum_{j=1}^d s_{ij} x_i y_j}{d \cdot \|\mathbf{x}\| \cdot \|\mathbf{y}\|}. \quad (3)$$

Equation (3) is used for computing the similarity between a novice word and a sample word in the proposed version of the KNN algorithm.

Let us make some remarks on the encoding process the process of encoding words into numerical vectors and computing the similarity between them. A word is encoded into a numerical vector by the feature candidate extraction, the feature selection, and the feature value assignment. The similarity between features which are given as texts is computed by equation (1). The similarity between numerical vectors which represent words is computed, considering the feature similarities by equation (3). In future, we will consider computing the similarities among some features rather than all features, for improving the computation speed.

B. Feature Similarity based KNN Algorithm

This section is concerned with the KNN algorithm which considers the feature similarities. The version was initially proposed as the approach to the text classification by Jo in 2019 [3]. The similarity metric between numerical vectors which was described in the previous section is used for computing the similarity between a novice vector and a training example. The labels of its nearest neighbors are voted with their identical weights for decoding the label of a novice vector. This section is intended to describe the proposed version of the KNN algorithm.

The process of computing the similarity between a novice item and a training example is illustrated in Figure 3. The labeled words which are gathered as samples are encoded into numerical vectors, and they are notated by a set $Tr = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$. A novice word is encoded into a numerical vector notated by $\mathbf{x}$. Its similarities with the numerical vectors which represent the sample words are computed by equation (3), as $\text{sim}(\mathbf{x}, \mathbf{x}_1), \text{sim}(\mathbf{x}, \mathbf{x}_2), \dots, \text{sim}(\mathbf{x}, \mathbf{x}_N)$. Some training examples which are most similar as the novice input are selected as its nearest neighbors.

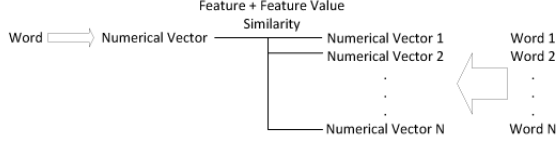


Figure 3. Similarities of Novice Input with Training Examples

In Figure 4, the process of selecting nearest neighbors by the ranking selection is illustrated. The training examples are sorted by the descending order of their similarities, after computing their similarities with a novice example. The training examples with their higher similarities with a novice example are selected as its nearest neighbors, as expressed in equation (4),

$$Nr = Select_k(Tr, \mathbf{x}), Nr \subseteq Tr \quad (4)$$

The set of nearest neighbors, Nr , is composed with elements as expressed in equation (5),

$$Nr = \{\mathbf{x}_1^{near}, \mathbf{x}_2^{near}, \dots, \mathbf{x}_k^{near}\} \quad (5)$$

The elements in the set, Nr , are used for voting their labels in the next step.

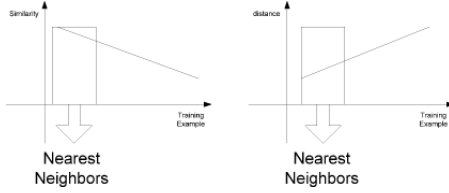


Figure 4. Selection of Most Similar or Least Distant Training Examples as Nearest Neighbors

The process of voting the labels of nearest neighbors for decoding the label of a novice word is illustrated in Figure 5. The predefined categories are notated by $c_1, c_2, \dots, c_m$, and the label of a nearest neighbor is notated by $c_j = label(\mathbf{x}_i^{near})$. The number of nearest neighbors which belong to the category, c_j , is notated by $Count(Nr, c_j)$. The label of the novice item, $\mathbf{x}$, is decided by equation (6),

$$label(\mathbf{x}) = \underset{j=1}{\operatorname{argmax}}^m Count(Nr, c_j) \quad (6)$$

The process of deciding the label of a novice item by equation (6), is called voting.

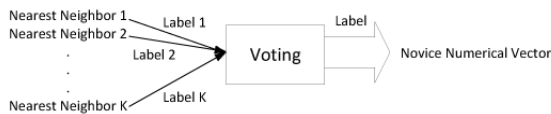


Figure 5. Voting of Labels of Nearest Neighbors for Deciding Label of Novice Input

Let us make some remarks on the proposed version of KNN algorithm. It computes the similarities of a novice item

with the training examples. The training examples with their highest similarities with the novice item are selected as its nearest neighbors. The label of the novice item is decided by voting the labels of its nearest neighbors. The ranking selection is adopted for selecting the nearest neighbors from training examples, in this version.

C. System Architecture

This section is concerned with the keyword extraction system which adopts the KNN algorithm which considers the feature similarities. In Section II-B, we described the proposed version of KNN algorithm as the approach to the keyword extraction. It is viewed as a binary classification which classifies a word into keyword or non-keyword. This section is intended to describe the keyword extraction system with respect to its function and its architecture.

The process of collecting the sample words and encoding them into numerical vectors for implementing the keyword extraction system. The keyword extraction is viewed as a binary classification which classifies a word into keyword or non-keyword is illustrated in Figure 6. The topic-based word classification belongs to the domain independent classification where a same word is always classified identically with regardless of the domain, whereas the keyword extraction is the domain dependent classification where a same word may be classified differently depending on the domain. The words which are labeled with keyword or non-keyword are collected domain by domain. It is required to tag the input text with its own domain for executing the keyword extraction.

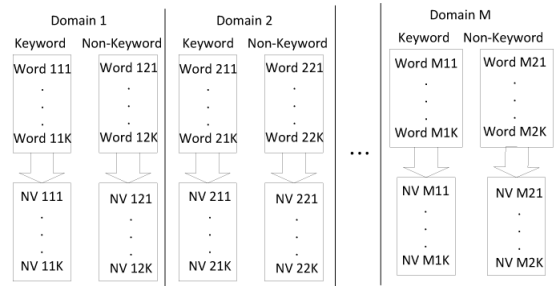


Figure 6. Preparation of Training Examples

The entire architecture of the proposed keyword extraction system is illustrated in Figure 7. A text is given as the input, and words are extracted from it in the indexing module. In the encoding module, sample words in the keyword group and the non-keyword group and ones which are indexed from the text are mapped into numerical vectors. The words which are indexed from the text are classified into one of the two categories in the similarity computation module and the voting module. The system consists of the four modules: indexing module, encoding module, similarity computation module, and voting module.

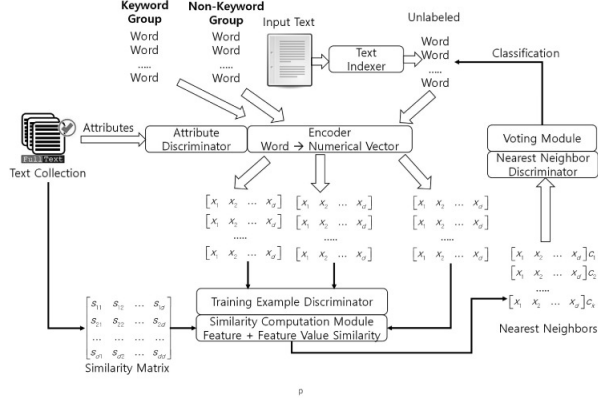


Figure 7. System Architecture

The execution process of the proposed system is illustrated as a block diagram in Figure 8. Sample words which are labeled with keyword or non-keyword are collected within a domain, and they are encoded into numerical vectors. An input text is indexed into a list of words, and they are also encoded into numerical vectors. The nearest neighbors are selected by the similarity computation, and the label is decided by voting ones of their nearest neighbors for each word. The words which are classified into keyword are extracted as the final output.

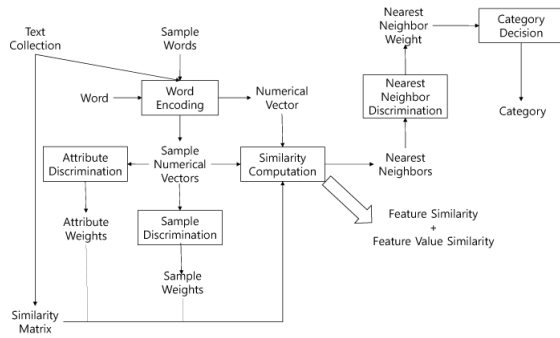


Figure 8. System Flow

Let us make some remarks on the proposed system which is illustrated in Figure 7 as its architecture. We propose the similarity metric which considers both the feature similarity and the feature value similarity. The poor discrimination among numerical vectors which is caused by their sparse distributions is solved. The classification performance is expected to be improved by what is proposed in this research. In the next research, we present the graphical user interface and the source code which are necessary for implementing the system as the complete version.

D. Connection with Text Clustering

This section is concerned with the connection of the keyword extraction with the text clustering. In the previ-

ous section, we mentioned the keyword extraction as an instance of domain dependent classification. The domains are automatically constructed from the text collection by clustering texts. The AHC algorithm which is proposed in [2] is used for implementing the text clustering. This section is intended to describe the keyword extraction system which is connected with the text clustering for automating the construction of domains.

The architecture of the text clustering system which was proposed in [2] is illustrated in Figure 9. The texts in the group are encoded into numerical vectors, and the AHC algorithm which is proposed in [2] is adopted as the approach. It starts with singletons as many as items, computes the similarities of all possible cluster pairs, and merge the cluster pair with its highest similarity into one cluster. The two steps are iterated until the number of clusters reach the desired number. We may consider replacing the AHC algorithm by its variants for improving the speed and the performance.

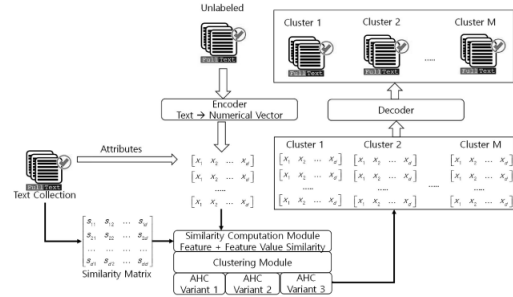


Figure 9. Text Clustering System

The connection of the keyword extraction with the text clustering is illustrated in Figure 10. The M domains are defined by clustering texts in a group, initially and automatically. A novice text is given as the input, and its domain, domain D , is decided by its similarities with domains. A classifier which corresponds to domain D is nominated and classify each word in a novice text into keyword or non-keyword. The M keyword extraction systems are viewed as independent ones, each of which classifies each word into one of the two categories.

The execution flow of keyword extraction which is connected with the text clustering is illustrated in Figure 11. Unlabeled texts are clustered into subgroups, each of which is a domain. Sample words which are labeled with keyword or non-keyword are generated from each domain. These sample words are provided for training the classifier in each keyword extraction system. Each classifier is used for judging whether each word which is indexed from the input text is keyword or not.

Let us make some remarks on the connection of the keyword extraction with the text clustering. It is the process of segmenting a text group into subgroups based on their

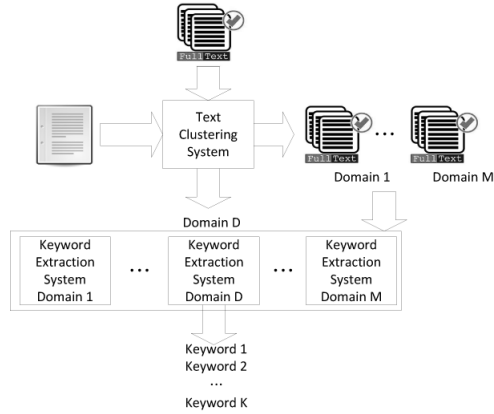


Figure 10. Keyword Extraction System connected with Text Clustering

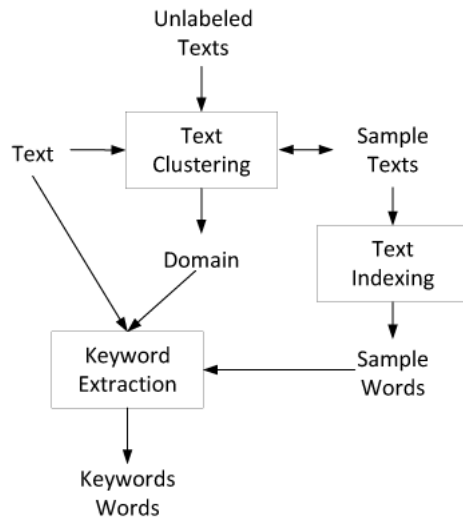


Figure 11. System Flow of Keyword Extraction connected with Text Clustering

content-based similarities. The role of text clustering is to define the domains initially in the architecture of connecting the keyword extraction with the text clustering. In each domain, sample words are generated, and its corresponding classifier is trained with them. In the future research, we consider using the keyword extraction for clustering texts in the opposite direction.

REFERENCES

- [1] T. Jo, "Extracting Keywords from News Articles using Feature Similarity based K Nearest Neighbor", 68-71, The Proceedings of International Conference on Information and Knowledge Engineering, 2018.
- [2] T. Jo, "Clustering Texts using Feature Similarity based AHC Algorithm", 5993-6003, Journal of Intelligent and Fuzzy Systems, 5, 2018.
- [3] T. Jo, "Text Classification using Feature Similarity based K Nearest Neighbor", 13-21, AS Medical Science, 3, 4, 2019.